UNIVERSITÀ DI PISA

PRINCIPLES FOR SOFTWARE COMPOSITION

ROBERTO BRUNI

Anno accademico	2022/23
CdS	INFORMATICA
Codice	375AA
CFU	9

Moduli	Settore/i	Tipo	Ore	Docente/i
MODELLI DI CALCOLO	INF/01	LEZIONI	72	ROBERTO BRUNI

Obiettivi di apprendimento

Conoscenze

Gli studenti dovranno acquisire solide basi di conoscenza circa i principi della semantica operazionale e denotazionale, le tecniche per relazionare l'una con l'altra nel caso di linguaggi imperativi e funzionali di ordine superiore. Inoltre dovranno familiarizzare con i principi della semantica operazionale e astratta per algebra di processi concorrenti, con le principali logiche temporali e modali (HM, LTL, CTL*, mu-calculus) usate per esprimere proprietà di tali processi e con modelli operazionali con probabilità. Gli ultimi saranno studiati dal punto di vista degli automi probabilistici e stocastici, delle catene di Markov e dal linguaggio di modellazione PEPA.

Modalità di verifica delle conoscenze

In caso di didattica a distanza, gli esami scritti potranno essere sostituiti da test di auto-valutazione e/o progetti software individuali.
L'esame scritto servirà a verificare che gli studenti abbiano compreso il materiale presentato durante il corso e che siano in grado di organizzare e strutturare i concetti appresi in efficaci risposte scritte.
Durante l'esame orale gli studenti dovranno dimostrare di aver compreso gli aspetti più concettuali e di essere in grado di argomentare le risposte con proprietà di linguaggio usando i termini corretti.
Metodi di verifica:

- Esame orale finale
- Esame scritto finale
- Compitini

Informazione aggiuntiva:
La valutazione sarà basata su esami scritti e orale. L'esame scritto finale non sarà necessario se i compitini saranno valutati positivamente.

Capacità

Al termine del corso gli studenti dovrebbero essere in grado di dimostrare semplici proprietà formali di linguaggi e sistemi imperativi, dichiarativi, concorrenti e probabilistici. In particolare dovrebbero essere in grado di applicare correttamente i più comuni principi di induzione e la teoria del punto fisso.

Modalità di verifica delle capacità

In caso di didattica a distanza, gli esami scritti potranno essere sostituiti da test di auto-valutazione e/o progetti software individuali.
L'esame scritto consiste di esercizi che richiedono l'applicazione di tecniche formali di prova ad alcuni casi specifici accuratamente selezionati.
L'esame orale verte principalmente sulle definizioni e sulle proprietà mostrate nel corso, incluse le dimostrazioni dei risultati principali.

Comportamenti

Alla fine del corso gli studenti dovrebbero essere in grado di comprendere e applicare sistemi logici e definizioni induttive, di definire la semantica di linguaggi di programmazione, e di ragionare su eventuali equivalenze e corrispondenze tra specifiche astratte.

Modalità di verifica dei comportamenti

Esame scritto e orale.

In caso di didattica a distanza, gli esami scritti potranno essere sostituiti da test di auto-valutazione e/o progetti software individuali.

Prerequisiti (conoscenze iniziali)

Non ci sono prerequisiti stringenti, ma ci si aspetta che gli studenti abbiano una buona familiarità con la matematica discreta, con le formule logiche del prim'ordine, con le grammatiche libere da contesto e siano in grado di comprendere frammenti di codice scritti in linguaggi imperativi e funzionali.

Indicazioni metodologiche

Lezioni frontali e esercitazioni collettive.
Attività di apprendimento:

- partecipazione attiva alle lezioni
- domande e discussioni col docente
- assegnamenti di esericizi a fine lezione, da dscutere nella lezione successiva
- studio individuale e a gruppi

Frequenza: fortemente consigliata

Programma (contenuti dell'insegnamento)

Introduzione di cinque modelli computazionali differenti e studio delle loro proprietà formali principali (imperativo: IMP, funzionale: HOFL, processi concorrenti: CCS, calcoli con nomi: pi-calculus, sistemi probabilistici e stocastici: Segala automata e PEPA) accompagnati da principi di induzione e da metodi di dimostrazione.

Bibliografia e materiale didattico

Testo principale:

- Roberto Bruni, Ugo Montanari, “Models of Computation”, Springer Texts in Computer Science, 2017.

Letture opzionali consigliate:

- Glynn Winskel, “The formal Semantics of Programming Languages”, MIT Press, 1993. Chapters: 1.3, 2, 3, 4, 5, 8, 11.
  “La Semantica Formale dei Linguaggi di Programmazione”, traduzione italiana a cura di Franco Turini, UTET 1999.
- Robin Milner, “Communication and Concurrency”, Prentice Hall, 1989. Chapters: 1-7, 10.
- Luca Aceto et al., Reactive Systems, Cambridge University Press, 2011. Chapters 1-7.

Indicazioni per non frequentanti

Nessuna variazione.

Modalità d'esame

In caso di didattica a distanza, gli esami scritti potranno essere sostituiti da test di auto-valutazione e/o progetti software individuali.
Esame scritto (due compitini o un esame finale) e esame orale.
L'esame scritto consiste di una serie di esercizi da risolvere in 2 ore. Ogni esercizio ha un punteggio assegnato e il voto finale è la somma dei punteggi conseguiti sui singoli esercizi. I testi delle sessioni di esame precedenti sono raggiungibili dalla pagina del corso.
L'esame orale prende spunto dalla discussione dell'esame scritto per poi affrontare gli argomenti principali presentati durante il corso. La durata tipica dell'esame orale è 30'/40'.

Altri riferimenti web

Alcuni strumenti presentati nel corso:

- Haskell, https://www.haskell.org/
- Erlang, http://www.erlang.org/
- Google Go, http://golang.org/
- CAAL, http://caal.cs.aau.dk/
- PEPA http://www.dcs.ed.ac.uk/pepa/

Ultimo aggiornamento 29/07/2022 11:15